D. Rescue Nibel!

time limit per test

2 seconds

memory limit per test

256 megabytes

input

standard input

output

standard output

Ori and Sein have overcome many difficult challenges. They finally lit the Shrouded Lantern and found Gumon Seal, the key to the Forlorn Ruins. When they tried to open the door to the ruins... nothing happened.

Ori was very surprised, but Sein gave the explanation quickly: clever Gumon decided to make an additional defence for the door.

There are nn lamps with Spirit Tree's light. Sein knows the time of turning on and off for the ii-th lamp — lili and riri respectively. To open the door you have to choose kk lamps in such a way that there will be a moment of time when they all will be turned on.

While Sein decides which of the kk lamps to pick, Ori is interested: how many ways there are to pick such kk lamps that the door will open? It may happen that Sein may be wrong and there are no such kk lamps. The answer might be large, so print it modulo 998244353998244353.

**Input**

First line contains two integers nn and kk (1≤n≤3⋅1051≤n≤3⋅105, 1≤k≤n1≤k≤n) — total number of lamps and the number of lamps that must be turned on simultaneously.

Next nn lines contain two integers lili ans riri (1≤li≤ri≤1091≤li≤ri≤109) — period of time when ii-th lamp is turned on.

**Output**

Print one integer — the answer to the task modulo 998244353998244353.

**Examples**

**input**

**Copy**

7 3

1 7

3 8

4 5

6 7

1 3

5 10

8 9

**output**

**Copy**

9

**input**

**Copy**

3 1

1 1

2 2

3 3

**output**

**Copy**

3

**input**

**Copy**

3 2

1 1

2 2

3 3

**output**

**Copy**

0

**input**

**Copy**

3 3

1 3

2 3

3 3

**output**

**Copy**

1

**input**

**Copy**

5 2

1 3

2 4

3 5

4 6

5 7

**output**

**Copy**

7

**Note**

In first test case there are nine sets of kk lamps: (1,2,3)(1,2,3), (1,2,4)(1,2,4), (1,2,5)(1,2,5), (1,2,6)(1,2,6), (1,3,6)(1,3,6), (1,4,6)(1,4,6), (2,3,6)(2,3,6), (2,4,6)(2,4,6), (2,6,7)(2,6,7).

In second test case k=1k=1, so the answer is 3.

In third test case there are no such pairs of lamps.

In forth test case all lamps are turned on in a time 33, so the answer is 1.

In fifth test case there are seven sets of kk lamps: (1,2)(1,2), (1,3)(1,3), (2,3)(2,3), (2,4)(2,4), (3,4)(3,4), (3,5)(3,5), (4,5)(4,5).

By \* **Ashishgup**, contest: Codeforces Round #672 (Div. 2), problem: (D) Rescue Nibel!, **Accepted**, [#](https://codeforces.com/contest/1420/submission/93664393), Copy

1. **#include** <bits/stdc++.h>
2. **using** **namespace** std;
4. **#define** IOS ios::sync\_with\_stdio(0); cin.tie(0); cout.tie(0);
5. **#define** endl **"\n"**
6. **#define** **int** **long** **long**
8. **const** **int** N = 3e5 + 5;
9. **const** **int** MOD = 998244353;
11. **int** fact[N], invfact[N];
13. **int** pow(**int** a, **int** b, **int** m)
14. {
15. **int** ans=1;
16. **while**(b)
17. {
18. **if**(b&1)
19. ans=(ans\*a)%m;
20. b/=2;
21. a=(a\*a)%m;
22. }
23. **return** ans;
24. }
26. **int** modinv(**int** k)
27. {
28. **return** pow(k, MOD-2, MOD);
29. }
31. **void** precompute()
32. {
33. fact[0]=fact[1]=1;
34. **for**(**int** i=2;i<N;i++)
35. {
36. fact[i]=fact[i-1]\*i;
37. fact[i]%=MOD;
38. }
39. invfact[N-1]=modinv(fact[N-1]);
40. **for**(**int** i=N-2;i>=0;i--)
41. {
42. invfact[i]=invfact[i+1]\*(i+1);
43. invfact[i]%=MOD;
44. }
45. }
47. **int** nCr(**int** x, **int** y)
48. {
49. **if**(y>x)
50. **return** 0;
51. **int** num=fact[x];
52. num\*=invfact[y];
53. num%=MOD;
54. num\*=invfact[x-y];
55. num%=MOD;
56. **return** num;
57. }
59. **int** n, k;
60. **int** l[N], r[N];
61. **map**<**int**, **vector<int>** > event;
63. **int32\_t** main()
64. {
65. IOS;
66. precompute();
67. cin >> n >> k;
68. **for**(**int** i = 1; i <= n; i++)
69. {
70. cin >> l[i] >> r[i];
71. event[l[i]].push\_back(1);
72. event[r[i] + 1].push\_back(-1);
73. }
74. **int** active = 0, ans = 0;
75. **for**(**auto** &it:event)
76. {
77. **int** newCnt = 0;
78. **for**(**auto** &j:it.second)
79. **if**(j == -1)
80. active--;
81. **else**
82. newCnt++;
83. **for**(**int** i = 1; i <= min(k, newCnt); i++)
84. {
85. **if**(k - i > active)
86. **continue**;
87. **int** curVal = nCr(newCnt, i) \* nCr(active, k - i);
88. curVal %= MOD;
89. ans += curVal;
90. ans %= MOD;
91. }
92. active += newCnt;
93. }
94. cout << ans << endl;
95. **return** 0;
96. }